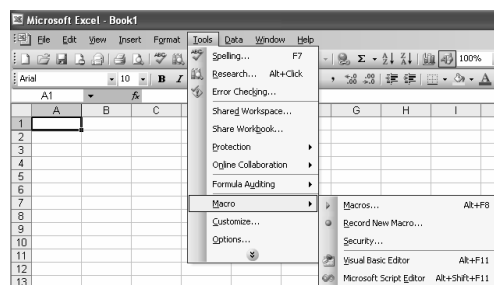
Mengenal Macro (Visual Basic for Application)

Macro, atau biasa juga dikenal dengan istilah Visual Basic for Application (VBA), merupakan pengembangan bahasa pemrograman Visual Basic yang diterapkan dalam aplikasi Excel. Berbeda dengan program pengembang Visual Basic, pemrograman yang dibuat menggunakan Macro hanya dapat dibangun dan digunakan pada aplikasi Excel. Program yang dibuat menggunakan Macro tidak dapat berjalan, sebelum Anda menjalankan Excel terlebih dahulu. Pemrograman Macro Excel mempunyai beberapa keuntungan sebagai berikut.

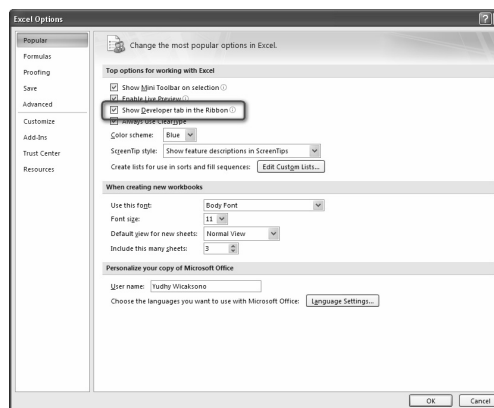
- **Menghemat waktu**. Penyelesaian pekerjaan menggunakan Macro lebih cepat dibandingkan cara manual, karena prosesnya dikerjakan secara otomatis.
- **Menghemat tenaga**. Selain menghemat waktu, penyelesaian pekerjaan menggunakan Macro juga dapat menghemat tenaga.
- **Mengurangi tingkat kesalahan**. Kemungkinan adanya kesalahan dalam menyelesaikan pekerjaan secara manual dapat saja terjadi, meskipun Anda seorang yang sangat ahli dalam menggunakan Excel. Penyelesaian pekerjaan menggunakan Macro secara konsisten akan menyelesaikan suatu pekerjaan berdasarkan perintah yang tertulis dalam kode Macro sehingga

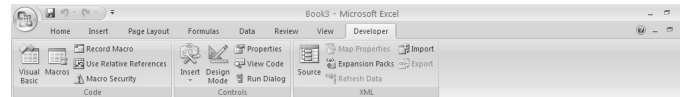
tingkat kesalahan yang mungkin timbul sangat kecil. Kesalahan hanya dapat terjadi jika ada kesalahan perintah pada kode Macro.

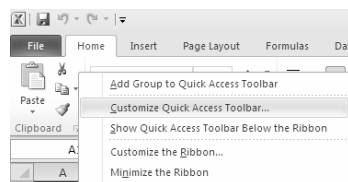
2.1 Penggunaan Fitur Macro

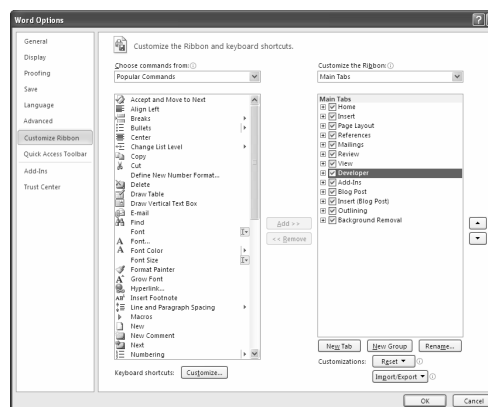
Penggunaan fitur Macro pada Excel 2003 dilakukan melalui menu **Macro** yang terdapat dalam menu utama **Tools**.

Gambar 2.1 Penggunaan fitur Macro Excel 2003.

Bagi pengguna Excel 2007 dan Excel 2010, Anda harus menampilkan tab Developer dalam Ribbon terlebih dahulu untuk menggunakan Macro.

1. Untuk pengguna Excel 2007, klik **Office Button** kemudian pilih **Excel Options**. Muncul kotak dialog Excel Options.

Gambar 2.2 Menampilkan tab Developer Excel 2007.

2. Pilih opsi **Popular**. Beri tanda contreng pada pilihan **Show Developer tab in the Ribbon** lalu klik tombol **OK**. Tampilan tab Developer dalam Ribbon terlihat seperti pada Gambar 2.3.

Gambar 2.3 Tab Developer dalam Ribbon.

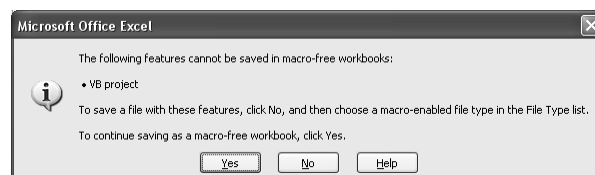
3. Untuk menampilkan tab Developer pada Ribbon Excel 2010, klik kanan area Ribbon kemudian pilih menu **Customize the Ribbon...** Muncul kotak dialog Excel Options pada pilihan Customize Ribbon.

Gambar 2.4 Menu klik kanan area Ribbon.

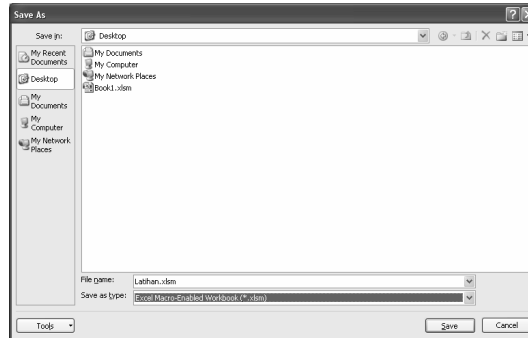
4. Anda juga dapat menampilkan kotak dialog Excel Options dengan cara klik tab **File** kemudian pilih **Options**. Muncul kotak dialog Excel Options. Pilih opsi **Customize Ribbon**.

Gambar 2.5 Menampilkan tab Developer Excel 2010.

5. Pilih **Main Tabs** pada kotak pilihan Customize the Ribbon:. Beri tanda contreng tab **Developer** dalam daftar di sebelah kanan. Klik tombol **OK**.

2.2 Format File

Workbook standar Excel 2003 dengan format file XLS mendukung fitur Macro, sedangkan workbook standar Excel 2007 dan Excel 2010 dengan format XLSX tidak mendukung Macro. Untuk menyimpan workbook Excel 2007 atau Excel 2010 yang mengandung Macro, Anda harus menggunakan format XLSM atau XLS (Excel 97-2003).

1. Klik tombol **Office Button** kemudian pilih menu **Save As** untuk menyimpan workbook. Bagi pengguna Excel 2010 klik tab **File** kemudian pilih menu **Save As**. Muncul kotak dialog Save As. Jika Anda menyimpan workbook yang mengandung Macro dengan format file XLSX, muncul kotak pesan seperti pada Gambar 2.6.

Gambar 2.6 Kotak pesan format penyimpanan workbook.

2. Klik tombol **Yes** jika workbook akan tetap disimpan dalam format file XLSX tanpa mengaktifkan Macro. Jika opsi ini yang Anda pilih, Macro dalam workbook akan dihapus. Untuk menyimpan workbook dengan Macro klik tombol **No**. Muncul kotak dialog Save As.
3. Ketikkan nama file pada kotak isian File name:. Pada kotak pilihan Save as type: pilih format file **Excel Macro-Enabled Workbook(*.xlsm)** atau **Excel 97-2003 Workbook(*.xls)** kemudian klik tombol **Save**.

Gambar 2.7 Kotak dialog Save As.

2.3 Keamanan Macro

Otomatisasi Macro Excel terkadang disalahgunakan oleh pengguna yang tidak bertanggung jawab untuk menyebarkan kode Macro yang berpotensi menimbulkan kerugian, misalnya kode Macro untuk menghapus data. Untuk mengantisipasi hal tersebut, Excel secara default akan memberi peringatan melalui kotak dialog jika pengguna membuka workbook yang mengandung Macro.

2.3.1 Excel 2003

Excel 2003 secara default akan menampilkan kotak dialog Security Warning jika pengguna membuka workbook yang di dalamnya terdapat Macro. Tampilan kotak dialog Security Warning terlihat seperti pada Gambar 2.8.

Gambar 2.8 Kotak dialog Security Warning.

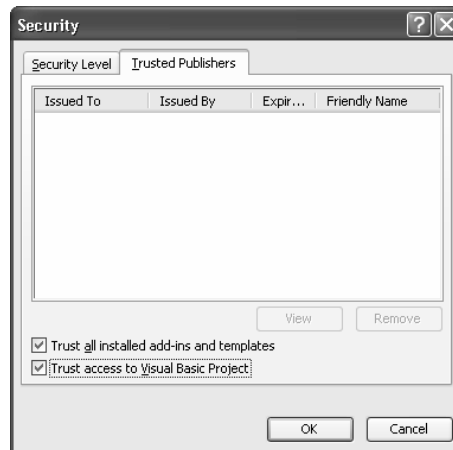
Kotak dialog tersebut memberi peringatan bahwa dalam workbook yang Anda buka terdapat Macro. Peringatan tersebut ditampilkan untuk mencegah dijalankannya Macro yang berbahaya. Jika Anda tidak mengenal Macro pada workbook tersebut, klik tombol **Disable Macros** untuk menonaktifkan Macro. Apabila Anda mengenal Macro dalam workbook dan memang berniat menggunakannya, klik tombol **Enable Macros** untuk mengaktifkan Macro.

Anda dapat melakukan pengaturan keamanan Macro melalui kotak dialog **Security**. Pilih menu **Tools** > **Macro** > **Security...** kemudian muncul kotak dialog Security.

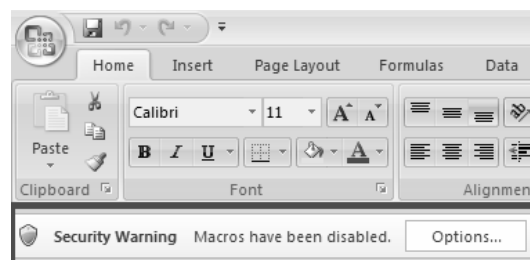
Gambar 2.9 Kotak dialog Security - tab Security Level.

Pilih tab **Security Level**. Pilihan pengaturan keamanan Macro yang dapat Anda pilih adalah sebagai berikut.

- **High** atau **Very High**. Apabila salah satu opsi ini dipilih, Macro yang ada dalam workbook tidak akan dijalankan ketika dibuka.
- **Medium**. Apabila opsi ini dipilih, Excel akan menampilkan kotak dialog Security Warning pada saat workbook yang mengandung Macro dibuka. Dijalankan atau tidaknya Macro akan tergantung pilihan Anda pada kotak dialog Security Warning.
- **Low**. Apabila opsi ini dipilih, Macro yang ada dalam workbook akan selalu dijalankan, tanpa melalui kotak dialog Security

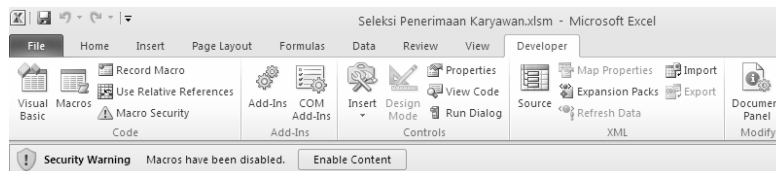
Warning. Jika Anda memilih opsi ini, pilih tab **Trusted Publishers** kemudian beri tanda contreng pada pilihan **Trust all installed add-ins and templates** dan **Trust access to Visual Basic Project**.

Gambar 2.10 Kotak dialog Security - tab Trusted Publishers.

2.3.2 Excel 2007 dan Excel 2010

Tampilan peringatan keamanan dan pengaturan keamanan Macro Excel 2007 dan Excel 2010 berbeda dengan tampilan peringatan keamanan dan pengaturan keamanan Macro Excel 2003. Excel 2007 secara otomatis akan menonaktifkan Macro saat workbook yang mengandung Macro pertama kali dibuka. Muncul peringatan keamanan seperti terlihat pada Gambar 2.11.

Gambar 2.11 Peringatan keamanan Macro Excel 2007.

Klik tombol **Options....** Muncul kotak dialog Microsoft Office Security Options. Untuk mengaktifkan Macro, pilih opsi **Enable this content** kemudian klik tombol **OK**.

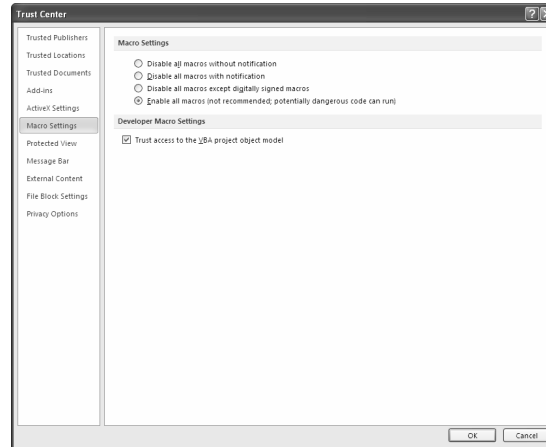
Gambar 2.12 Kotak dialog Microsoft Office Security Options.

Excel 2010 secara otomatis akan menonaktifkan Macro ketika workbook yang mengandung Macro dibuka. Muncul peringatan keamanan seperti terlihat pada Gambar 2.13. Klik tombol **Enable Content** untuk mengaktifkan Macro.

Gambar 2.13 Peringatan keamanan Macro Excel 2010.

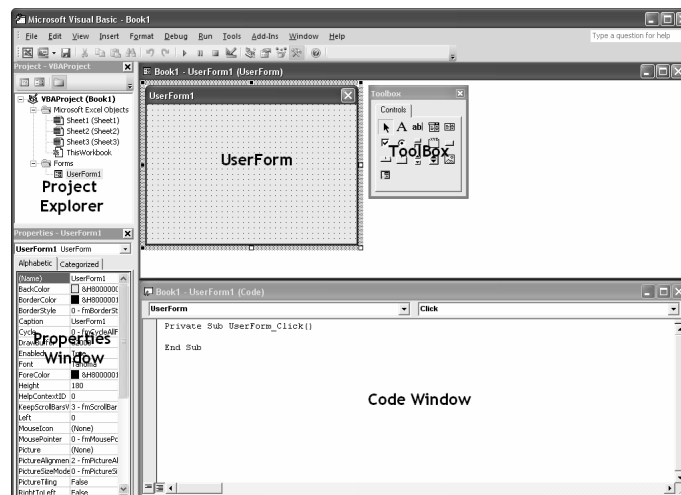
Excel 2010 selanjutnya akan menampilkan kotak dialog Microsoft Office Excel Security Notice jika Anda membuka workbook yang mengandung Macro. Apabila Anda tidak mengenal Macro pada workbook tersebut, klik tombol **Disable Macros** untuk menon-aktifkan Macro. Untuk mengaktifkan Macro, klik tombol **Enable Macros**.

Gambar 2.14 Kotak dialog Microsoft Office Excel Security Notice.

Untuk mengatur keamanan Macro, klik tombol **Macro Security** dalam tab Developer group Code. Muncul kotak dialog Trust Center pada pilihan Macro Settings. Anda dapat melakukan pengaturan keamanan Macro sebagai berikut.

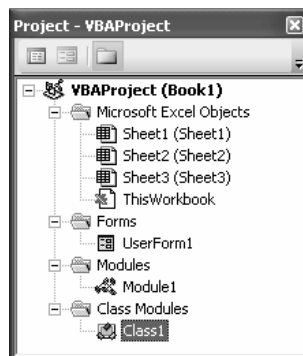
- **Disable all macros without notification**. Apabila opsi ini dipilih, Macro dalam workbook tidak akan dijalankan ketika dibuka.
- **Disable all macros except digitally signed macros**. Jika opsi ini dipilih, Macro dalam sebuah workbook tidak akan dijalankan ketika dibuka, kecuali untuk Macro yang dikembangkan oleh pengembang terpercaya.
- **Disable all macros with notification**. Apabila opsi ini dipilih, Excel akan menampilkan kotak dialog Microsoft Office Excel Security Notice ketika Anda membuka workbook yang mengandung Macro. Dijalankan atau tidaknya sebuah Macro selanjutnya tergantung pada pilihan Anda.
- **Enabled all macros (not recommended; potentially dangerous code can run)**. Apabila opsi ini dipilih, Macro yang ada dalam workbook akan selalu dijalankan, tanpa melalui kotak dialog Microsoft Office Excel Security Notice. Opsi ini tidak direkomendasikan, terutama untuk macro dalam workbook yang tidak Anda kenal, karena berpotensi menimbulkan kerugian. Jika Anda tetap memilih opsi ini, beri tanda contreng pada pilihan **Trust access to the VBA project object model**.

Gambar 2.15 Kotak dialog Trust Center.

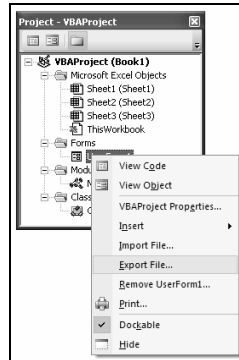
2.4 Visual Basic Editor

Visual Basic Editor merupakan lingkungan kerja, tempat di mana Macro Excel dibuat. Tampilan Visual Basic Editor sangat berbeda dengan tampilan utama Excel.

Gambar 2.16 Tampilan Visual Basic Editor.

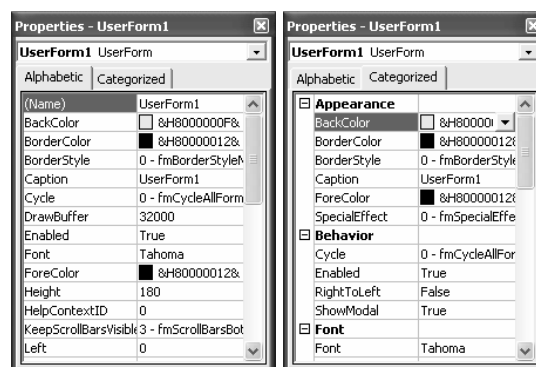
Pilih menu **Tools** > **Macro** > **Visual Basic Editor** untuk menampilkan Visual Basic Editor pada Excel 2003. Untuk menampilkan Visual Basic Editor pada Excel 2007 atau Excel 2010, klik tombol **Visual Basic** dalam tab Developer group Code. Visual Basic Editor juga dapat ditampilkan menggunakan kombinasi tombol **Alt+F11** pada keyboard.

2.4.1 Project Explorer

Project Explorer digunakan untuk melakukan navigasi terhadap seluruh objek yang ada dalam proyek VBA sebuah workbook. Secara garis besar, objek dikelompokkan ke dalam **Microsoft Excel Objects**, **Forms**, **Modules,** dan **Class Modules**. Untuk menampilkan atau mengaktifkan Project Explorer, pilih menu **View** > **Project Explorer** (atau tekan kombinasi **Ctrl+R**). Anda juga dapat menampilkan Project Explorer dengan cara klik tombol **Project Explorer** yang terdapat dalam toolbar Standard.

Gambar 2.17 Tampilan Project Explorer.

Melalui Project Explorer, Anda dapat menampilkan kode Macro, menampilkan objek, mengatur properti VBA, menambahkan objek, mencetak objek dan kode yang terdapat di dalamnya, serta melakukan impor dan ekspor objek. Untuk menggunakan fitur tersebut, klik kanan Project Explorer kemudian muncul daftar menu seperti terlihat pada Gambar 2.18.

Gambar 2.18 Daftar menu klik kanan Project Explorer.

2.4.2 Window Properties

Window Properties digunakan untuk menampilkan properti yang dimiliki objek. Untuk menampilkan atau mengaktifkan window Properties, pilih menu **View** > **Properties Window** atau klik tombol **Properties Window** (icon) pada toolbar Standard. Cara yang sama juga dapat dilakukan dengan menekan tombol **F4** pada keyboard.

Gambar 2.19 Tampilan window Properties.

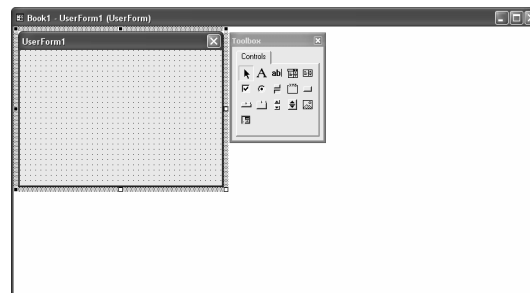
Properti objek pada window Properties dapat dilihat berdasarkan urutan abjad (**Alphabetic**) ataupun bedasarkan kategori (**Categorized**). Window Properties secara otomatis akan menampilkan properti objek yang sedang aktif (terpilih).

2.4.3 Window Code

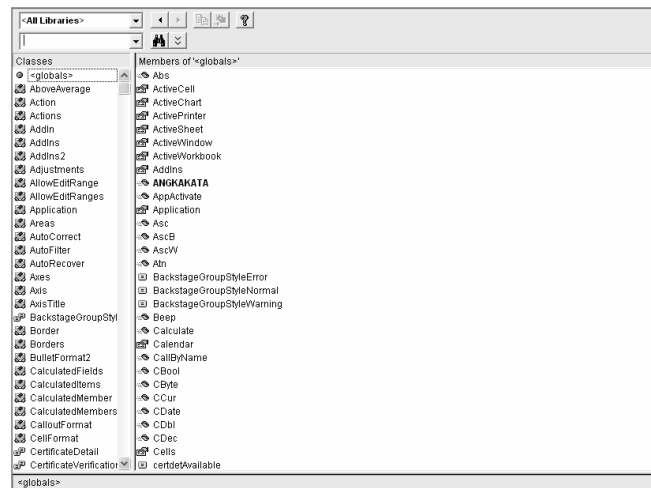
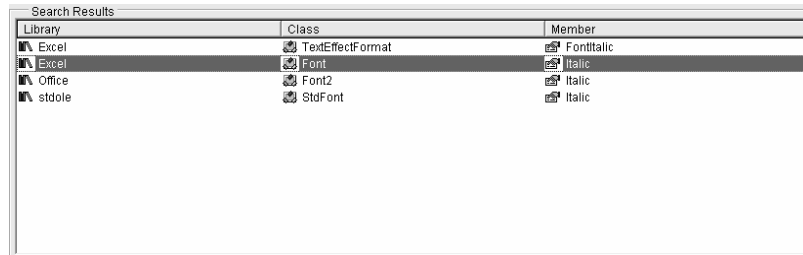
Window Code digunakan untuk melihat, membuat atau melakukan modifikasi kode Macro. Pada window Code, terdapat kotak pilihan Object Selector dan Event Selector. Kotak pilihan Object Selector digunakan untuk memilih objek atau koleksi objek yang akan ditampilkan kode Macro-nya, sedangkan kotak pilihan Event Selector digunakan untuk memilih event pada objek terpilih.

Gambar 2.20 Tampilan Window Code.

2.4.4 Window Object

Window Object merupakan tempat yang digunakan untuk menampilkan, membuat, dan mengatur objek UserForm serta objek kontrol dalam UserForm secara visual. Untuk menampilkan window Object, pilih menu **View** > **Object**. Anda juga dapat menampilkan window Object dengan menekan kombinasi tombol **Shift+F7** pada keyboard.

Gambar 2.21 Tampilan Window Object.

2.4.5 Object Browser

Object Browser digunakan untuk menampilkan atau melakukan pencarian terhadap semua objek, koleksi objek, properti, method, atau event yang terdapat dalam VBA.

Gambar 2.22 Tampilan Object Browser.

Untuk menampilkan Object Browser, pilih menu **View** > **Object Browser**, atau tekan tombol **F2**. Anda juga dapat menampilkan Object Browser dengan cara klik tombol **Object Browser** yang terdapat dalam toolbar Standard. Untuk mencari informasi objek, properti, method, atau event tertentu menggunakan Object Browser, lakukan langkah-langkah sebagai berikut:

1. Ketikkan objek, properti, method, atau event yang akan Anda cari pada kotak pilihan Search Text. Dalam contoh kali ini, ketikkan **Italic** untuk mencari informasi mengenai properti Italic.

Gambar 2.23 Menggunakan fasilitas pencarian.

2. Klik tombol **Search** atau tekan tombol **Enter** pada keyboard. Hasil pencarian kemudian ditampilkan seperti terlihat pada Gambar 2.24.

Search Results

Library	Class	Member
Excel	TextEffectFormat	FontItalic
Excel	Font	Italic
Office	Font2	Italic
stdole	StdFont	Italic

Gambar 2.24 Tampilan hasil pencarian.

3. Klik salah satu item hasil pencarian yang Anda inginkan, misalnya pilih properti **Italic** pada Library Excel dan Class Font. Informasi properti yang Anda pilih akan ditampilkan di bagian pojok kiri bawah Object Browser.
4. Untuk meng-*copy* properti yang Anda pilih, klik tombol **Copy to Clipboard**. Tekan kombinasi **Ctrl+V** untuk menyalin properti yang sudah Anda *copy*.

Object Browser menggunakan ikon yang berbeda untuk memudahkan Anda dalam membedakan objek, properti, method, atau event. Berikut ikon yang digunakan Object Browser untuk membedakan objek, properti, method, atau event.

- ikon objek.
- ikon properti.
- ikon event.
- ikon method.
- ikon predifened constant, yaitu konstanta yang secara default sudah disediakan. Konstanta untuk VBA diawali dengan prefiks vb, misalnya vbYes. Untuk Excel, konstanta diawali dengan prefiks xl, misalnya xlChart.

2.5 Kode Macro

Kode Macro merupakan serangkaian tulisan perintah yang akan dilaksanakan ketika Macro dijalankan. Kode Macro akan mengontrol dan menentukan dijalankannya sebuah Macro. Kode Macro dapat ditulis pada objek worksheet, workbook, Module, UserForm, atau Class Module.

2.5.1 Memenggal Kode Macro

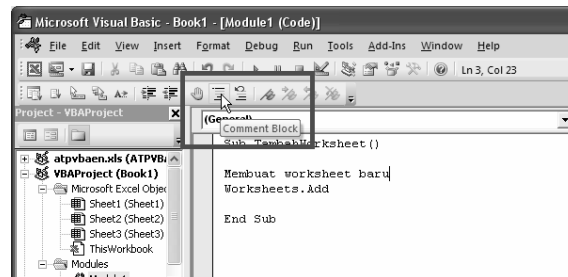
Kode Macro dapat dipenggal ke baris berikutnya, jika Anda merasa kode Macro terlalu panjang. Anda juga dapat melakukan pemenggalan pada komentar. Pemenggalan dilakukan dengan spasi yang diikuti garis bawah (_). Perhatikan contoh kode Macro yang dipenggal berikut:

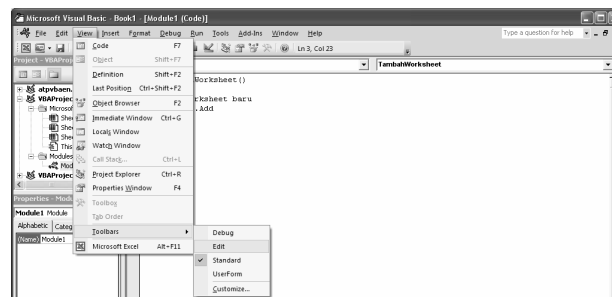
```
MsgBox "Nama worksheet sudah ada atau belum diisi", _
vbOKOnly + vbCritical, "Error Nama Worksheet"
```

2.5.2 Komentar

Anda dapat menambahkan komentar untuk memberi keterangan pada baris kode Macro tertentu. Komentar dapat ditambahkan pada suatu baris dengan menuliskan tanda petik satu (‘) di depan statement yang ingin dinyatakan sebagai komentar. Komentar tidak dianggap sebagai perintah sehingga tidak akan dijalankan. Perhatikan contoh berikut ini:

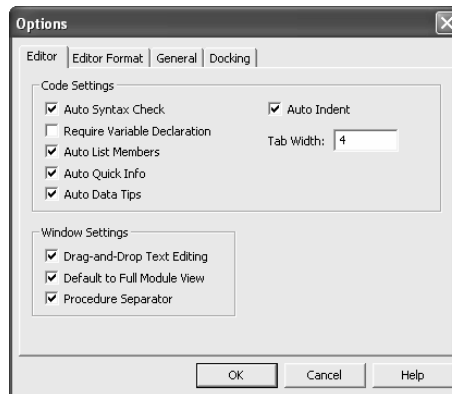
```
'Membuat worksheet baru
Worksheets.Add
```

`’Membuat worksheet baru` dianggap bukan perintah, melainkan komentar sehingga tidak akan dijalankan. **`Worksheets.Add`** dianggap sebagai perintah sehingga akan dijalankan. Untuk membuat komentar dengan mudah, pilih atau blok baris kode yang akan dibuat menjadi komentar kemudian klik ikon **Comment Block** pada toolbar Edit.

Gambar 2.25 Menambahkan komentar.

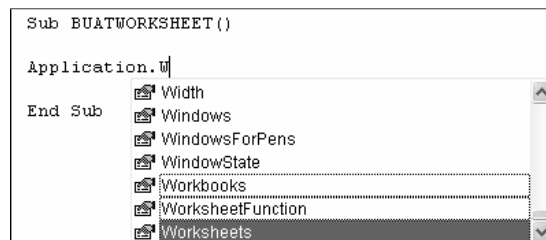
Apabila toolbar Edit belum ditampilkan, pilih menu **View** > **Toolbar** > **Edit** untuk menampilkan toolbar Edit.

Gambar 2.26 Menampilkan toolbar Edit.

2.5.3 Fitur Auto List Members

Saat menuliskan kode Macro, Anda dapat memanfaatkan fitur Auto List Members, yaitu fitur yang akan menampilkan daftar objek, koleksi objek, properti, atau method yang dimiliki sebuah objek. Dengan menggunakan fitur Auto List Members, kesalahan penulisan objek, koleksi objek, properti, ataupun method dapat diminimalkan. VBA secara default mengaktifkan fitur Auto List Members. Apabila tidak aktif, Anda dapat mengaktifkannya melalui kotak dialog Options. Berikut langkah-langkah pengaturan fitur Auto List Members:

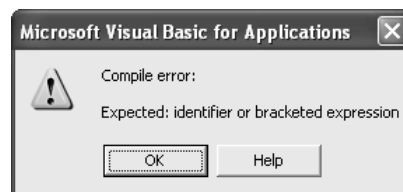
1. Pilih menu **Tools** > **Options...** kemudian muncul kotak dialog Options. Pilih tab **Editor**.

Gambar 2.27 Kotak dialog Options – tab Editor.

2. Beri tanda contreng pada pilihan Auto List Members kemudian klik tombol **OK**.

Untuk menampilkan fitur Auto List Members tekan kombinasi **Ctrl+J**. Daftar objek, koleksi objek, properti, atau metode yang dimiliki objek akan terlihat seperti pada Gambar 2.28. Anda dapat menekan tombol **Esc** pada keyboard untuk menyembunyikan fitur Auto List Members. Untuk memilih salah satu opsi yang ditampilkan daftar Auto List Members, tekan tombol **Tab** pada keyboard, atau klik ganda opsi yang ingin Anda pilih.

Gambar 2.28 Tampilan fitur Auto List Members.

2.5.4 Kesalahan Penulisan Kode Macro

Apabila kode Macro yang Anda tulis salah, VBA secara default akan menampilkan kotak pesan peringatan. Klik tombol **OK** jika

Anda mengetahui solusi untuk menangani kesalahan yang terjadi. Jika Anda memerlukan informasi bantuan untuk menangani kesalahan yang terjadi, klik tombol **Help**. VBA secara default mengaktifkan fitur ini. Jika tidak aktif, Anda dapat mengaktifkannya melalui kotak dialog Options tab Editor pilihan Auto Syntax Check.

Gambar 2.29 Kesalahan penulisan kode Macro.

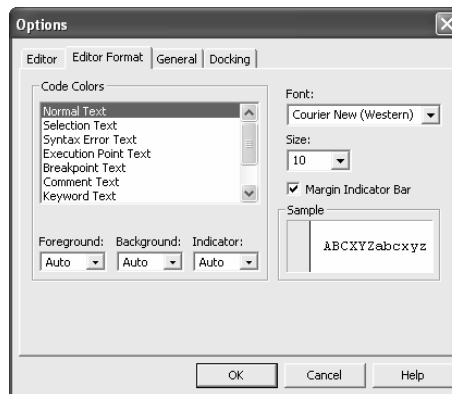
2.5.5 Fitur Auto Quick Info

Fitur Auto Quick Info digunakan untuk menampilkan informasi argumen dari fungsi, properti, atau method. VBA secara default mengaktifkan fitur Auto Quick Info. Apabila tidak aktif, Anda dapat mengaktifkannya melalui kotak dialog Options tab Editor pilihan Auto Quick Info.

Gambar 2.30 Informasi yang ditampilkan Auto Quick Info.

2.5.6 Pengaturan Format Font Kode Macro

Secara subjektif, penulis merasa nyaman dengan pengaturan default format font kode Macro. Jika Anda kurang merasa nyaman dengan pengaturan default format font kode Macro, Anda dapat melakukan pengaturan format font kode Macro sendiri.

1. Pilih menu **Tools** > **Options...** kemudian muncul kotak dialog Options. Pilih tab **Editor Format**.

Gambar 2.31 Kotak dialog Options – tab Editor Format.

2. Berikut pilihan pengaturan format font kode Macro yang dapat Anda terapkan:

   - **Code Colors**. Pada pilihan ini, Anda dapat mengatur warna font kode Macro (Foreground), background teks kode Macro (Background) atau indikator kode Macro (Indicator) elemen kode Macro pada daftar pilihan yang tersedia, misalnya **Selection Text**.
   - **Font**. Pada kotak pilihan Font: Anda dapat mengatur jenis font elemen kode Macro pada daftar pilihan yang tersedia.
   - **Size**. Pada kotak pilihan Size: Anda dapat mengatur ukuran font elemen kode Macro pada daftar pilihan yang tersedia.
   - **Margin Indicator Bar**. Beri tanda contreng pada pilihan ini untuk menampilkan indikator visual saat *debugging* Macro.

2.6 Objek

Objek merupakan komponen dalam Macro yang mempunyai properti dan metode sehingga dapat diubah atau dimanipulasi. Suatu objek dapat berupa satu objek atau kumpulan beberapa objek (koleksi objek). Sebuah koleksi juga merupakan objek

sehingga mempunyai properti dan metode yang dapat diubah (dimanipulasi). Objek dalam Macro mempunyai tingkatan dari yang paling umum sampai tingkatan yang paling khusus. Excel merupakan objek yang paling umum (Application) yang mempunyai tingkatan objek di bawahnya sebagai koleksi objek, misalnya Workbook, WorksheetFunction, dan koleksi objek lainnya.

Saat menuliskan kode Macro, objek dan koleksi objek dipisahkan menggunakan tanda titik (.). Untuk kondisi tertentu, Anda dapat menuliskan objek tanpa perlu menyebutkan objek tingkatan di atasnya, misalnya kode Macro untuk menyeleksi worksheet **Data** sebagai berikut:

```
Sheets("Data").Select
```

Apabila dalam waktu bersamaan terdapat dua buah workbook yang dibuka, di mana masing-masing workbook mempunyai worksheet Data, maka Anda harus menyebutkan objek tingkatan di atasnya (workbook) dalam kode Macro. Jika tidak disebutkan, workbook yang dipilih adalah workbook yang sedang aktif. Berikut contoh kode Macro yang harus Anda tuliskan jika workbook yang Anda maksud adalah Analisis.xls:

```
Workbooks("Analisis.xls").Sheets("Data").Select
```

2.7 Properti

Properti merupakan karekteristik yang dimiliki objek. Seperti halnya benda yang memiliki beberapa identitas, suatu objek juga memiliki beberapa properti yang menunjukkan identitas objek tersebut. Apabila suatu benda memiliki identitas panjang, lebar, tebal, warna, dan identitas lainnya, maka objek memiliki beberapa properti yang menunjukkan identitas, misalnya Name, Height, Width, BackColor, atau Caption. Pengaturan properti objek sangat penting untuk membedakan satu objek dengan objek yang lain. Properti objek dapat diatur melalui window Properties atau melalui kode Macro saat *runtime* (Macro dijalankan). Untuk mengubah properti objek melalui window Properties, pastikan objek dalam keadaan terpilih. Window Properties akan menampilkan daftar properti objek yang dipilih. Properti objek juga dapat diubah saat

runtime menggunakan kode Macro. Saat menuliskan kode Macro, objek dan properti dipisahkan menggunakan tanda titik (.). Anda dapat memanfaatkan fitur Auto List Members untuk menampilkan daftar properti sebuah objek.

2.8 Method

Method merupakan suatu set perintah, seperti halnya Function Procedure dan Sub Procedure, tetapi sudah tersedia di dalam suatu objek. Penggunaan method dalam kode program Anda, tergantung pada kaitan perintah dan jumlah argumen yang diperlukan, serta apakah method tersebut mengembalikan suatu nilai.

2.9 Variabel

Variabel merupakan tempat dalam memori komputer yang diberi nama sebagai pengenal dan dialokasikan untuk menampung data. Sesuai data yang ditampung, variabel harus mempunyai tipe data yang sesuai dengan isinya. Secara default, tipe data yang digunakan dalam variabel adalah variant. Jika Anda tidak mengetahui dengan pasti tipe data yang akan digunakan, tipe data sebaiknya dikosongkan. Deklarasi variabel harus diletakkan sebelum baris perintah yang menggunakan variabel tersebut.

Dalam mendeklarasikan variabel, Anda perlu memerhatikan jangkauan variabel tersebut. Jangkauan variabel pada VBA dapat diketahui dengan kata kunci **Public**, **Private** dan **Dim** pada saat variabel dideklarasikan. Variabel yang dideklarasikan dengan kata kunci **Public** akan tersedia bagi semua Procedure di semua Module dalam suatu proyek, tempat di mana variabel tersebut dideklarasikan. Variabel yang dideklarasikan dengan kata kunci **Private** akan tersedia bagi semua Procedure dalam Module, tempat di mana variabel tersebut dideklarasikan. Variabel yang dideklarasikan dengan kata kunci **Dim** hanya akan tersedia dalam Procedure, tempat di mana variabel tersebut dideklarasikan. Format kode Macro dalam mendeklarasikan variabel adalah sebagai berikut:

```
Jangkauan NamaVariabel As TipeData
```

Berikut contoh pendeklarasian sebuah variabel:

```
Public Umur As Single
Private Nama As String
Dim DataProduksi As Range
```

2.10 Konstanta

Konstanta adalah nama yang menyimpan suatu nilai yang tidak dapat berubah. Kecepatan proses pada konstanta lebih cepat dibandingkan variabel karena tidak perlu menunggu pengisian data. Dalam mendeklarasikan kontanta, Anda perlu memerhatikan jangkauan konstanta tersebut. Jangkauan konstanta pada VBA dapat diketahui dengan kata kunci **Public**, **Private,** dan **Const** pada saat konstanta dideklarasikan. Konstanta yang dideklarasikan dengan kata kunci **Public** akan tersedia bagi semua Procedure di semua Module dalam suatu proyek, tempat di mana konstanta tersebut dideklarasikan. Konstanta yang dideklarasikan dengan kata kunci **Private** akan tersedia bagi semua Procedure dalam Module, tempat di mana konstanta tersebut dideklarasikan. Konstanta yang dideklarasikan dengan kata kunci **Const** hanya akan tersedia dalam Procedure, tempat di mana konstanta tersebut dideklarasikan. Format kode Macro untuk mendeklarasikan konstanta adalah sebagai berikut:

```
Jangkauan NamaKontsanta As TipeData = Ekspresi
```

Berikut contoh pendeklarasian sebuah konstanta:

```
Public Const Berat As Integer = 50
Private Const Nama As String = "Yudhy Wicaksono"
Const Penjualan As Integer = 25000
```

2.11 Array

Array merupakan variabel yang mampu menyimpan beberapa nilai dengan tipe data yang sama. Kumpulan nilai tersebut satu sama lain dibedakan dengan indeks dan masing-masing disebut elemen array. Beberapa nilai data yang mempunyai tipe data sama akan lebih mudah jika dimasukkan ke dalam sebuah array dibandingkan dimasukkan dalam beberapa variabel yang berbeda. Ada dua jenis array, yaitu *fixed array* dan *dynamic array*. *Fixed*

array merupakan array yang ukurannya tetap. Format kode Macro dalam mendeklarasikan *fixed array* adalah sebagai berikut:

```
Jangkauan NamaArray(Indeks) As TipeData
```

Indeks merupakan jumlah elemen yang akan digunakan pada array. Anda dapat menuliskan sebuah *fixed array* dengan cara berikut ini:

```
Dim harga As Single
Nama harga = Array(2500, 3500, 5000)
```

Anda juga dapat menuliskan *fixed array* dengan cara berikut ini:

```
Dim harga(2) As Single
harga(0) = 2500
harga(1) = 3500
harga(2) = 5000
```

Sebuah array dibatasi oleh batas bawah dan batas atas. Secara default, batas bawah array adalah nol (0). Apabila batas bawah array adalah nol (0), maka pada Indeks, Anda isikan dengan jumlah elemen dikurangi satu. Misalnya Anda akan menggunakan 5 elemen array, maka pada saat mendeklarasikan array, **Indeks** Anda isikan dengan angka 4. Selain dengan menggunakan batas bawah nol (0), Anda juga dapat mengubah batas bawah array dengan angka satu (1). Berikut cara yang dapat Anda lakukan untuk mendeklarasikan batas bawah array:

1. Menggunakan pernyataan **Option Base**. Format kode Macro dalam menggunakan **Option Base** adalah sebagai berikut:

   ```
   Option Base 1
   ```

   Misalnya, Anda ingin menentukan batas bawah array adalah 1, maka sebelum pendeklarasian array, Anda tuliskan kode Macro:

   ```
   Option Base 1
   Dim harga(3) As String
   ```

Anda tidak perlu menuliskan pernyataan Option Base, apabila Anda menggunakan nol (0) sebagai batas bawah array.

2. Menggunakan pernyataan **To**. Format kode Macro dalam menggunakan pernyataan **To** adalah sebagai berikut:

```
KataKunci NamaArray(BatasBawah To BatasAtas) As TipeData
```

Misalnya, Anda ingin menentukan batas bawah array adalah 1 dan batas atas array adalah 3, maka kode Macro yang Anda tuliskan untuk mendeklarasikan array adalah sebagai berikut:

```
Dim harga(1 To 3) As String
```

Berbeda dengan *fixed array*, ukuran dalam *dynamic array* dapat berubah. *Dynamic array* sangat berguna dalam pemrograman yang jumlah elemen *array*nya tidak bisa diketahui sejak awal. Format kode Macro dalam mendeklarasikan *dynamic array* adalah sebagai berikut:

```
KataKunci NamaArray() As TipeData
```

Berikut contoh penulisan sebuah *dynamic array*:

```
Dim harga() As Single
```

2.12 Tipe Data

Setiap variabel, konstanta, atau array yang dideklarasikan dalam VBA mempunyai tipe data. Pemilihan tipe data akan menentukan nilai apa yang dapat ditampung oleh variabel, konstanta, atau array. Oleh karena itu, pemilihan tipe data harus tepat dan sesuai dengan nilai yang akan ditampung. Secara default, tipe data yang digunakan dalam variabel, konstanta, atau array adalah variant. Jika Anda tidak mengetahui dengan pasti tipe data yang akan digunakan, maka tipe data sebaiknya dikosongkan atau diisi dengan tipe data variant. Berikut beberapa tipe data dalam VBA.

Tipe data	Ukuran	Contoh
Integer	2 byte	Semua bilangan antara -32.768 sampai 32.767
Long	4 byte	Semua bilangan antara -2.147.483.648 sampai 2.147.483.648
Single	4 byte	Bilangan negatif antara -3,402823 x 10^{38} sampai -1,401298 x 10^{-45} Bilangan positif antara 1,401298 x 10^{-45} sampai -3,402823 x 10^{38}
Double	8 byte	Bilangan negatif antara -1,7976931348623 x 10^{308} sampai -4,940656458623 x 10^{-324} Bilangan positif antara 4,940656458623 x 10^{-324} sampai 1,7976931348623 x 10^{308}
Decimal	14 byte	±79228162514264337593543950335 (tanpa titik desimal) ±7,9228162514264337593543950335 (dengan 28 angka di belakang titik desimal)
Currency	8 byte	Bilangan dengan nilai antara -922.337.203.685.477,5808 sampai 922.337.203.685.477,5808
String	1 byte	Untuk menyimpan teks berisi 0 sampai 2 milyar karakter
Byte	1 byte	Bilangan antara 0 sampai 255
Boolean	2 byte	Berisi nilai True (benar) atau False (salah)
Date	8 byte	Menyimpan informasi tanggal dan waktu. Tanggal antara 1 Januari 100 sampai 31 Desember 9999. Waktu antara 00:00:00 sampai 23:59:59.
Object	4 byte	Digunakan untuk mengakses objek apa saja yang diperlukan oleh VBA, disimpan dalam alamat memory objek tersebut.
Variant	16 byte	Seluruh tipe data yang ada. Jika berupa teks, maka akan disimpan dalam bentuk teks. Jika berupa bilangan, maka akan disimpan dalam tipe Double.

2.13 Struktur Kontrol Keputusan

Visual Basic for Application menyediakan fasilitas untuk pengambilan keputusan berdasarkan kondisi tertentu, yaitu **If…Then**, **If…Then…Else** dan pernyataan **Select Case**.

2.13.1 If…Then

Pernyataan **If…Then** memungkinkan Anda untuk menjalankan sebuah statement atau beberapa statement apabila suatu kondisi terpenuhi. Format kode Macro untuk pernyataan **If…Then** adalah sebagai berikut:

```
If Kondisi Then

Statement

End If
```

Apabila kondisi terpenuhi, statement yang mengikuti Then akan dijalankan. Namun sebaliknya, jika kondisi tidak terpenuhi, maka statement yang mengikuti Then tidak akan dijalankan.

2.13.2 If…Then…Else

Pernyataan **If…Then…Else** digunakan untuk mendefinisikan beberapa blok statement yang akan dijalankan salah satu berdasarkan kondisi yang memenuhi syarat. Format kode Macro untuk pernyataan **If…Then…Else** adalah sebagai berikut:

```
If Kondisi1 Then

Statement1

ElseIf Kondisi2 Then

Statement2

Else

Statement3

End If
```

VBA pada awalnya akan menguji **Kondisi1**. Jika Kondisi1 terpenuhi, maka Statement1 akan dijalankan. Apabila tidak terpenuhi, maka VBA akan menguji **Kondisi2,** dan seterusnya sampai ditemukan suatu kondisi yang memenuhi syarat untuk menjalankan Statement.

2.13.3 Select Case

Pernyataan **Select Case** digunakan untuk menguji "ekspresi" pada setiap pernyataan Select Case. Format kode Macro untuk pernyataan Select Case adalah sebagai berikut:

```
Select Case Ekspresi

Case Ekspresi1

Statement1

Case Ekspresi2

Statement2

Case Ekspresi3

Statement3

End Select
```

2.14 Struktur Pengulangan

Struktur pengulangan dalam VBA adalah sebuah struktur yang menjalankan beberapa statement secara berulang-ulang. VBA mempunyai struktur pengulangan **Do…Loop**, **For…Next**, **While…Wend,** dan **For Each…Next**.

2.14.1 Do…Loop

Struktur pengulangan **Do…Loop** digunakan untuk menjalankan satu atau beberapa pernyataan jika kondisi benar atau hingga suatu kondisi tersebut menjadi benar. Kata kunci untuk struktur pengulangan yang menjalankan statement jika kondisi benar adalah **While**. Format kode Macro struktur Do…Loop dengan kata kunci While adalah sebagai berikut:

```
Do While Kondisi

Statement

Loop
```

Kata kunci untuk struktur pengulangan yang menjalankan statement hingga suatu kondisi tersebut menjadi salah adalah **Until**. Format kode Macro struktur Do…Loop dengan kata kunci Until adalah sebagai berikut:

```
Do Until Kondisi

Statement

Loop
```

2.14.2 For…Next

Struktur pengulangan **For…Next** digunakan untuk menjalankan satu atau beberapa statement dengan frekuensi pengulangan yang telah ditentukan. Struktur pengulangan For…Next digunakan untuk kondisi yang mempunyai nilai berurutan dan variabelnya mempunyai nilai numerik. Format kode Macro struktur For…Next adalah sebagai berikut:

```
For Variabel = NilaiAwal To NilaiAkhir

Statement

Next Variabel
```

Anda juga dapat menempatkan struktur pengulangan For…Next di dalam struktur pengulangan For…Next yang lain, atau bisa disebut ***Nested For***. Format kode Macro struktur *Nested For* adalah sebagai berikut:

```
For Variabel1 = NilaiAwal1 To NilaiAkhir1

For Variabel2 = NilaiAwal2 To NilaiAkhir2

Statement

Next Variabel2

Next Variabel1
```

2.14.3 While…Wend

Struktur pengulangan **While…Wend** digunakan untuk menjalankan satu atau beberapa statement selama suatu kondisi itu benar. Format kode Macro struktur While…Wend adalah sebagai berikut:

```
While Kondisi

Statement

Wend
```

Apabila kondisi benar, maka semua statement akan dijalankan dan ketika mencapai baris Wend, kontrol akan kembali lagi ke baris While untuk melakukan evaluasi kembali nilai dari kondisi. Apabila nilai kondisi masih memenuhi syarat atau benar maka proses pengulangan akan terjadi lagi sampai kondisinya salah. Apabila kondisi salah, maka program akan dihentikan oleh Wend.

2.14.4 For Each…Next

Struktur pengulangan **For Each…Next** merupakan struktur pengulangan elemen objek dalam group objek. Struktur pengulangan ini akan sangat membantu jika Anda tidak mengetahui berapa elemen dalam group yang akan diulang. Format kode Macro struktur For Each…Next adalah sebagai berikut:

```
For Each Elemen in Group

Statement

Next Elemen
```

2.15 End

Statement End dipakai untuk memaksa program berhenti dari suatu Sub Procedure, Function Procedure, ekspresi If, atau deklarasi With.

2.15.1 End Sub

End Sub digunakan untuk mengakhiri sebuah Sub Procedure. Perhatikan contoh berikut:

```
Sub BuatWorksheet()

Worksheets.Add

End Sub
```

2.15.2 End Function

End Function digunakan untuk mengakhiri sebuah Function Procedure. Perhatikan contoh berikut ini:

```
Function LUAS(panjang, lebar)

LUAS = panjang * lebar

End Function
```

2.15.3 End If

End If digunakan untuk mengakhiri penggunaan ekspresi If.

```
If Range("A1").Value < 5 Then
Range("A2").Value = "Nilai kurang"
End If
```

2.15.4 End With

End With digunakan jika kita ingin mengakhiri penggunaan With di awal sebuah pendeklarasian. Perintah With dan End With dapat digunakan untuk menyingkat suatu penulisan objek yang berulang-ulang.

```
Range("D6:F6").Font.Name = "Calibri"
Range("D6:F6").Font.FontStyle = "Italic"
Range("D6:F6").Font.Size = 12
Range("D6:F6").Font.Underline = xlUnderlineStyleNone
```

Contoh Sub Procedure berikut ini juga akan menghasilkan perintah yang sama dengan Sub Procedure di atas:

```
Range("D6:F6").Select

With Selection.Font
.Name = "Calibri"
```

```
.Font.FontStyle = "Italic"
.Font.Size = 12
.Font.Underline = xlUnderlineStyleNone
End With
```